VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

MACHINE LEARNING (CO3117) - CC01

Machine Learning
Course Assignment

Advisor: Nguyễn Đức Dũng
Students: Nguyễn Quốc Việt - 1953096

HO CHI MINH CITY, DEC 2021

Contents

1 Introduction

1.1 Face identity recognition

Face identity recognition or face recognition in short is an advance technology that is able to identify the person through photos, videos or any visual media elements. It is a biometric method performed by measuring parts in the face to determine the uniqueness data of each person with their face and facial expression.

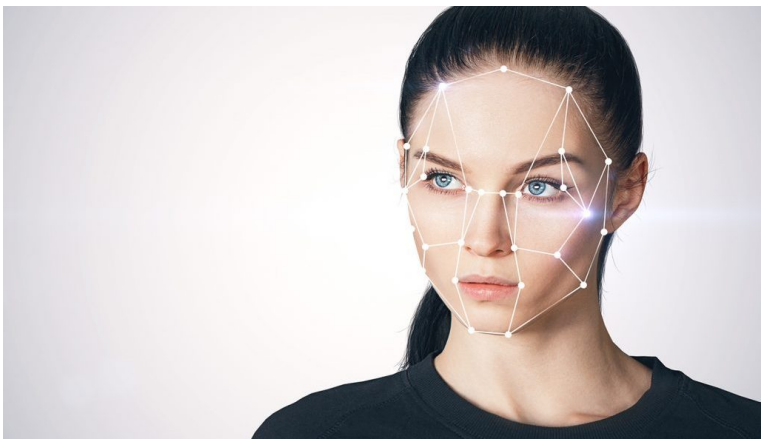

Figure 1: Face landmark for recognition

A face identity recognition systems works by capturing an incoming image from camera or picture, then this image will be compared with some relevant information that stored in the database. To be more precise, it analyses mathematically the image, calculates distances, extracts feature vectors... and feeds into a network to get the probability of the person the face belongs to or by an easier method, searching on database of tree by the feature vectors and find out who it belongs to.

1.2 Deep learning

Deep learning or in particular in our problem, the deep convolutional neural networks (CNN), has received increasing interest in face recognition, and several deep learning methods have been proposed.

Deep learning technology has reshaped the research landscape of face recognition since 2014, launched with the breakthroughs of DeepFace and DeepID methods. Since that landmark, deep face recognition techniques, which leverage the hierarchical architecture to learn discriminative face representation, have dramatically improved state-of-the-art performance and fostered numerous successful real-world applications. Deep learning applies multiple processing layers to learn representations of images with multiple levels of feature extraction.

2 Approach to solve Face Identity Recognition problem

2.1 CNN Background

2.1.1 Convolutional layer

Convolutional layers are the layers where filters are applied to the original image, or to other feature maps in a deep CNN. This is where most of the user-specified parameters are in the network. The most important parameters are the number of kernels and the size of the kernels.

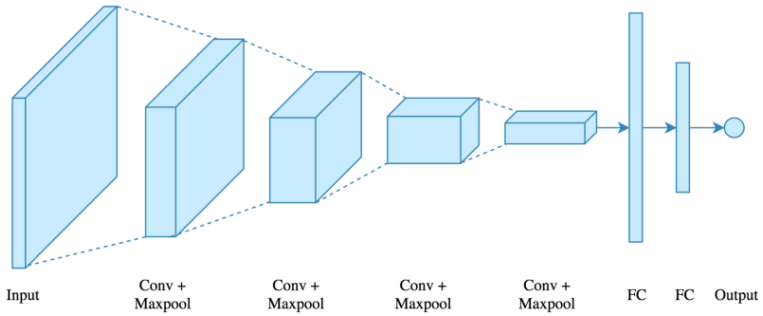

Figure 2: CNN model with layers

2.1.2 Pooling layer

Pooling layers are similar to convolutional layers, but they perform a specific function such as max pooling, which takes the maximum value in a certain filter region, or average pooling, which takes the average value in a filter region to reduce dimensionality.

2.1.3 Fully connected layers

Fully connected layers are placed before the classification output of a CNN and are used to flatten the results before classification. This is similar to the output layer of an MLP.

2.1.4 Activation function

Activation functions are mainly used to achieve non-linear variations in the neural network model. A linear activation function lacks to perform back propagation and hence it is not recommended to use in neural networks. The adequacy of activation function lies in making the system learn and execute difficult tasks; thereby the neural network becomes more powerful with non-linear activation functions.

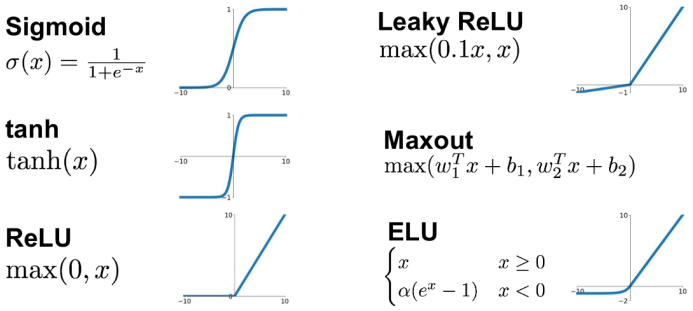

Figure 3: Popular activation functions

2.2 Method

To perform face identity recognition, an image must be passed as an input. The image may contain only the face that is well-cropped, but for those that is a capture image with background..., the image need to be cropped into a rectangle that only has the facial expression.

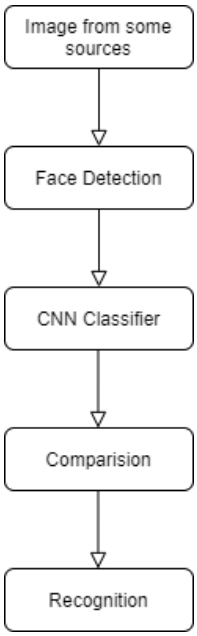

Figure 4: Face recognition flowchart

2.2.1 Frontal Face Detection

According to dlib's github page, dlib is a toolkit for making real world machine learning and data analysis applications in C++. While the library is originally written in C++, it has good, easy to use Python bindings. dlib can be used for face detection and facial landmark detection. The frontal face detector in dlib works really well. It is simple and just works out of the box.

The get_frontal_face_detector function does not accept any parameters. A call to it returns the pre-trained HOG + Linear SVM face detector.

Dlib's HOG + Linear SVM face detector is fast and efficient. By nature of how the Histogram of Oriented Gradients (HOG) descriptor works, it is not invariant to changes in rotation and viewing angle.

For more robust face detection, you can use the MMOD CNN face detector, available via the cnn_face_detection_model_v1 function. This method accepts a single parameter, modelPath, which is the path to the pre-trained mmod_human_face_detector.dat file residing on disk.

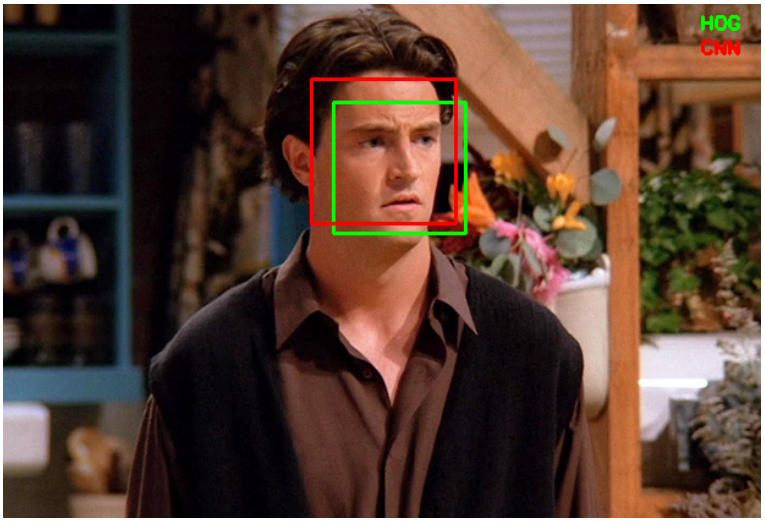

Figure 5: dlib Face Detection

2.2.2 CNN Model for the problem

The Figure 6 show the architecture of the CNN model to solve Face Identity Recognition problem. This is a simple model for the dataset of only 3 people that is: Harry Potter, Barack Obama and Andrew Ng.

The first layer will be the input layer, after the step of Face Detection, the face will then be convert to Grayscale image, the difference between RGB color space and Grayscale is that Grayscale has only 1 color channel, the features still stay on the image without any changes, this make the dimension of our input is smaller, yet still enough information. The size of image is 128×128, an usual size when feed into CNN so that Convolutional layer and pooling layer can capture full information.

The second layer is the Convolution2D layer from keras library. The image will be apply by a number of filters to extract the information then, feed into a ReLU function to take only the positive value, ignore the negative of gradient.

The third layer MaxPooling2D will act as the dimensionality reduction to reduce the number of calculations later on but still sustain the key features.

For the problem, there will be 2 Convolutional layer follow with ReLU and MaxPooling2D. After that, all is Flattening into fully connected layers. The output will be the number of people we are interested in classifying (3 for the problem). The Softmax layer will return the probability for each class, we will use numpy.argmax to get the max value to know which class the image belongs to.

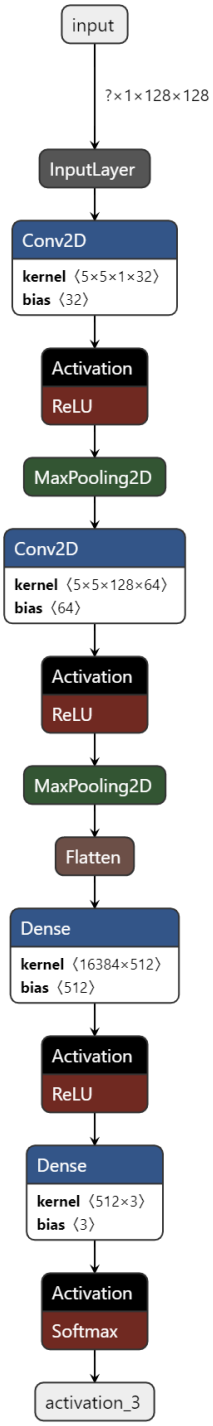

Figure 6: Model architecture

3 Implementation and Result

3.1 Dataset

I have configured my own dataset by looking for Youtube video and trim it into face-only video. The video then extract by cv2 into images contain the face, each person will have 200 images, form a dataset of 600 images. The dataset will be keep in the Google Drive link in the Notebook, but I also put the videos inside Github repo in case there are something wrong happen.

3.2 Source code and Instruction

The Github source code for the submission is: Face ID Recognition

To run the code, firstly we will need to check for the runtime, as dlib has troubles with performing on CPU, GPU for some trustful tools like Google Colab can be used instead of running on own machine.

Follow the pre-ordering codes in the Notebook, we will get the result at the last step.

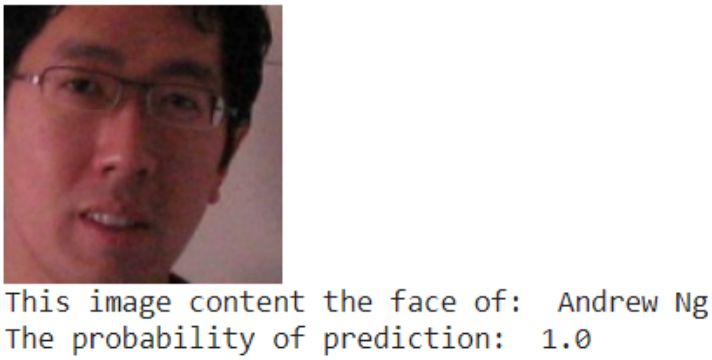

Figure 7: Prediction of model

3.3 Limitation

At the end of coding step and push it to Github repo, I realised that first, the dataset I have used is to small with only 3 people this result in the fact that overfitting happen, the model I implemented has not been apply by Regularization method, moreover, the similarity because of facial expression in near frame make the dataset has low distinctive.

4 Conclusion

In this project, I have proposed a simple CNN model for classifying 3 different people from video. The system evaluation with high Accuracy and low Loss score in the Notebook has achieved significant prediction although augmented data. This is just a small project for students with introduction level but can be later on for multiple problems in outside world, as a human lay his feet higher everyday, from simple to complicate insights.

References

[1] Mei Wang and Weihong Deng, "Deep Face Recognition: A Survey", *arXiv preprint arXiv:1804.06655*, 2018.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, "Deep Residual Learning for Image Recognition", *arXiv preprint arXiv:1512.03385*, 2015.

[3] Arnaldo P. Castaño. (2020, August 5). *Custom AI Face Recognition With Keras and CNN - CodeProject*. Retrieved 19:24, December 8, 2021, from https://www.codeproject.com/Articles/5275261/Custom-AI-Face-Recognition-With-Keras-and-CNN